A New Approach for the Validation of Conceptual Holonic Constructions

Javier Andrade, Juan Ares, Rafael García, María-Aurora Martínez,
Juan Pazos, Santiago Rodríguez, and Sonia Suárez

***Abstract*—The concepts of holon and holarchy were first applied in the manufacturing world to develop Holonic Manufacturing Systems. Since then, they have been used in many fields and have proved to be applicable concepts for developing applications in any business area. Resulting applications are based on conceptual holonic constructions. Like any model, a holarchy needs to be validated under real circumstances. Such validation assures the quality of the holarchy before it is implemented. In general, validation research tends to target: 1) the specific types of holons handled in each proposal and/or the selected development paradigms; and 2) algorithm performance rather than architecture quality. This paper proposes and evaluates a methodology that focuses on the quality of the architecture. This methodology is able to validate any holonic architecture built to meet trade requirements. Moreover, this is a general-purpose methodology. Therefore, the methodology would be valid for any domain and would not be invalidated by holon types and/or implementation paradigms emerging, changing or falling into disuse. For this purpose, we consider holonic architectures as conceptual models, using the pure holon and holarchy concepts and passing up not only any specific implementation paradigm but also any set of specific holon types.**

***Index Terms*—Conflict, discrepancy, holarchy, inconsistency, model, validation.**

I. Introduction

THE CONCEPTS of holon and holarchy were first used in the manufacturing industry. This was because the manufacturers were under increasing pressure due to customer demands for shorter lead times and higher product variety without concessions on product price or quality [1].

To satisfy the new requirements, manufacturing systems have had to deal with frequent process variations, as well as changes in production orders by means of constant adaptation and high flexibility. The manufacturing world has had to leave behind the structured and stable environments in which enterprises were operating and face highly changeable dynamic environments [2].

In this context, as indicated by Van Brussel *et al.* [1], neither hierarchical nor heterarchical systems work well. Hierarchical systems [3] follow the top-down method and strictly define components and functions. The components cannot act on their own initiative, as they only can communicate with their parent or descendant components. Thus, they have a typically rigid structure that prevents dynamic reactions to changes. Thanks to their complete autonomy, heterarchical systems [4] are able to communicate with any structural component, enabling efficient change management through continuous adaptation to the environment. Nevertheless, heterarchical control is quite complex and not easily applicable to industry.

The solution to this problem was a new method of organization: the holarchy. Holarchies combine the advantages of hierarchies and heterarchies. Since then, the use of holarchies has been encouraged in many areas, and they have proved to be a concept applicable for developing applications in any domain (see, e.g., [5]–[10]).

The neologism "holon" was coined by Arthur Koestler [11], following the holistic Aristotelian maxim where "the whole is more than the sum of its parts" [12] and also bearing in mind the parable of the two watchmakers [13]. The suffix "on," which means "part" or "particle," is added to "holos," the etymological origin of "holon," meaning the whole.

The term "holon" is therefore used to refer to entities that behave autonomously—as a whole—but are not self-sufficient, as they behave as a part of a bigger whole. Koestler referred to this phenomenon as the Janus effect, after the Roman mythological god who has two heads facing opposite directions. The down-directed face represents the whole, while the upper-directed face is the dependent part. The Janus effect is a key characteristic of any holon [6].

In this way, holons are able to establish a hierarchical/heterarchical structure: the holarchy. A holarchy can be defined as a system of self-regulated holons, which, depending on the circumstances, cooperate and/or collaborate in order to achieve a final goal. To sum up, a holarchy defines the basic and specific rules for the so-called collaboration of its holons, consequently restricting their autonomy and/or conditioning their decision-making processes.

According to the principle of encapsulation, a holarchy can be conceived and handled as a single holon by abstracting its internal composition. Likewise, a holarchy may be made up of several holarchies. Accordingly, holarchies are simultaneously wholes and parts.

For example, holarchies have been and still are being used in the manufacturing world to develop Holonic Manufacturing Systems (HMS). HMS are highly distributed systems based on autonomous and cooperative entities (holons), whose goal is to transform, transport, store, and/or validate information or physical objects [14]. Manufacturing is modeled by means of an assorted group of holons—e.g., the basic holons proposed by the PROSA guidelines [15], [16] or by ADACOR [17]—that cooperate to achieve the manufacturing goals. For examples of proposals in this field, see [18]–[24].

In virtual enterprises, a holarchy is a temporary coalition of distributed, autonomous, and also cooperative member enterprises (holons/holarchies) with an organizational architecture that has the distinctive characteristics of holonic systems [5].

Generally, the conceptual model and resulting system of any holonic application will be composed of a set of autonomous and non-self-sufficient holons/holarchies. However, there are no specific methods for implementing holonic systems in either the manufacturing world or any other field of holon application. Multi-agent technology is the most used tool [25]–[27]. Some researchers suggest [28] that this is because agents and holons are not unalike, and there are even some holonic multi-agent systems [29]. Agent-based alternatives for implementing holonic systems rely on multi-agent architectures like AGORA [30], [31] or ANEMONA [14] and also on negotiation protocols like Contract Net [32]. Remember, though, that holons and agents are not exactly the same thing [28].

Irrespective of how holonic systems are implemented, using either agents or other elements, the holonic architecture, like any other architecture, is a model of reality. Generally speaking then, the primary modeling tasks are: 1) build the model based on its requirements; and 2) validate that the model fits reality. Only after these tasks are complete should the model be deployed [33].

It follows from the applications of holons and holarchies that there has been plenty of research dealing with the construction and application of holonic models; however, as with any other model, validation is also a key task.

As indicated by Leitao *et al.* [34], validation is crucial for guaranteeing that the designed model suitably represents the specifications of the real systems, that is, that it is correct. Validation will determine whether the model is accepted or rejected. Not only does this step detect building faults but it also checks whether the model is a fair representation of the reality in the field of work concerned. Validation is the most important and difficult modeling activity, as a model that is found to be valid will be accepted, whereas we will have to go back to the drawing board if a model is found to be invalid.

In the following, we outline a small sample of significant examples to illustrate how holonic systems are currently being validated. Bal and Hashemipour [35] propose a virtual reality-based methodology for enhancing the design and implementation processes of holonic control systems in manufacturing practice. This methodology validates the implementation of the holonic control according to some qualitative, as well as quantitative, performance indicators through discrete event simulations with dynamic virtual reality interfaces. Blanc *et al.* [21] present a manufacturing execution system with a PROSA-based reference holon architecture [15], [16], and implemented using Java Agent Development framework [36]. The system is validated by means of a discrete-event simulator. Also, Leitao *et al.* [17], [34] present the ADACOR set of holon classes for HMS specification. It is implemented using agent technology, and the PASCELL software tool is used for quantitative/qualitative validation [37], [38].

On the whole, all the related (domain-dependent) proposals give instructions for validating and testing the system. However, validation focuses on the holon types and/or the selected implementation paradigms used by each proposal, and, as Van Brussel *et al.* [1] anticipated, the validation mechanisms test algorithm performance rather than the quality of the architecture. Therefore, their main goal is not to check that the model specifically fits all and no other than the requirements of reality.

Obviously, it is necessary to validate that the holonic model meets the requirements for which it is built, although it is equally important to check that the model meets no other than the specified requirements. In the manufacturing world—where these models were first developed—as in all other domains, time, funds, and resources are limited and have to be properly and efficiently managed. Therefore, when building a holonic system to satisfy certain trade requirements, it is important to make sure that no relationships are established involving responsibilities other than those strictly necessary, because, as a general rule, such responsibilities would entail time, costs, and/or resources that could be put to better use on other tasks.

The process for validating holonic architectures should, likewise, not only be independent of the holon types and the implementation paradigms used, but also be general enough for application to any holonic architecture in any domain. The resulting validation process would not then be invalidated by approaches for holonic systems construction emerging, changing or falling into disuse as time passes.

In summary, this paper presents a means for validating the quality of holonic architectures in conformance with specific trade requirements. The holarchy is thus viewed as a conceptual model. We also elude specific implementation paradigms, specific domains and any specific set of holon types by working with the pure concepts of holon and holarchy.

This paper is organized as follows. Section II presents the approach proposed for validating holonic architectures. Section III evaluates this proposal. Section IV presents a case study (from outside the manufacturing world, using holonic architectures) and, lastly, Section V reports our conclusions.

II. Proposed Approach

This paper proposes a Checkland-type methodology [39] for validating any holonic architecture built to satisfy a group of requirements or trade needs. This validation should focus on the quality of the architecture, as defined in Section I and not merely on operational behavior.

Another goal of the proposed methodology is that it should not be confined to any holon specification-related formalism (e.g., PROSA, ADACOR, etc.), as it is intended to be general enough to accommodate any type of current and future holons. Following the same principle, the proposed methodology is

also designed to elude any specific implementation paradigm for holonic systems, and particularly agent-based technology which is the most widespread nowadays. The proposed methodology is intended to be applicable, no matter what new implementation paradigms or holon types might appear in the future. To achieve this goal, we have to identify the properties or key features of holons and holarchies to define the pure concepts for use. These are properties that are independent of not only holon types but also specific implementations, enabling the holonic system to be handled as a conceptual model. After validation, the model could be implemented in later development stages using the most suitable paradigm at the time.

A. Generic Holon/Holarchy Properties

A holon has been defined as an element in its own right that represents an autonomous, cooperative, and self-organized behavior. Holons can also be organized in a hierarchical/heterarchical structure, called holarchy. A holarchy is a number of holons that cooperate to achieve common goals/objectives. A holarchy can also be conceived and managed as a single holon by abstracting its internal behavior. Therefore, holarchies are simultaneously wholes and parts.

In the last analysis, a holon/holarchy is an element capable of performing one or more given tasks. This way, an analogy can be drawn between the capabilities of a holon/holarchy and an individual's career skills. Personnel evaluation is widely used to detect and analyze these career skills. As skills are characteristics used to differentiate individuals, their correct measurement could lead to the prediction of the results of the performance of the tasks in which these individuals participate.

There are different types of skills. They can be relatively broad and used in a great many tasks (e.g., verbal ability, spatial visualization, numerical ability) or very specific and play a unique role in the performance of highly specialized tasks. The literature is rife with research and proposals along these lines [40], including, for instance, the Thurstone PMA battery [41] or the differential aptitude test battery [42].

Applying this analogy, a holon/holarchy is basically considered to have a number of abilities, some being specific to one specialized task, others being more general and related to more tasks. On this ground, a holon/holarchy can be said to have two key features: goals and associated abilities.

Goals are the specific objectives of a given holon/holarchy, meaning the specific activities that it is capable of performing (e.g., stock management). Abilities, on the other hand, are a holon/holarchy's additional goal-related capabilities (e.g., if its goal is to make a calculation, its associated ability will be to make that calculation rapidly). An associated ability does not exist separately; it is always linked to one or more goals and serves the purpose of improving or enhancing goal performance. Hence, goals are related to specific holon capabilities, whereas abilities are related to more general aptitudes that might be associated with such capabilities.

Similarly, a holarchy takes responsibility for the global goals, which it tackles them as partial tasks. These tasks will be carried out by other holons/holarchies. A given holon/holarchy will be allocated one of these partial tasks, provided it satisfies certain conditions. We have specified these conditions using the role concept proposed by Soriano [43].

TABLE I
CHARACTERISTICS OF HOLONS/HOLARCHIES AND ROLES

Property	Meaning
Identifier	Univocal element identification
Name	Main term that names the element
Synonyms	Other terms than might also name the element
Abbreviations	Abbreviations used to refer to the element
Description	Definition and/or description of the element
Support sources	Human and/or non-human sources that have additional knowledge for a better understanding of the element
Observations	Any additional observation about the element worth noting

According to Soriano, a holarchy comprises a group of roles, each representing one or more partial tasks of the holarchy. Moreover, a role could be part of more than one holarchy, as multiple higher ranking operations may obviously require performance of the same task. This characteristic clearly does not apply to simple holons.

Some properties of not only the holon/holarchy but also the role are defined for considering nomenclature and further details. These properties, together with their meaning, are shown in Table I.

B. Key Conceptual Modelling Concepts

A number of terms related to conceptual modeling should be defined to clarify the methodology proposed for the validation of holonic models. These terms are as follows [44].

1) Viewpoint. Incomplete information about not only the problem or need, but also the actual problem-solving process. In this respect, a viewpoint can be said to be an incomplete and particular approach/vision related both to the problem, the problem domain, and to the problem-solving process.
2) Discrepancy. A generic term used to refer to conflicts, as well as inconsistencies.
3) Conflict. The goals of different stakeholders interfere with each other, that is, two parts do not want to achieve the same thing.
4) Inconsistency. A rule establishing an imperative relationship between two elements is broken, that is, a given stated rule is not satisfied under a particular circumstance.

In this paper, we do not consider discrepancies to be an intolerable phenomenon that the modeller should systematically reject outright. On the contrary, they are considered as a routine thing that could be beneficial for the actual process of problem understanding and modeling [45].

This "positivist" approach to discrepancies is not at all new. In physics, specifically the physics of light, some phenomena are easier to explain from the viewpoint of waves, whereas others are more understandable from the viewpoint of particles. Although both views were considered to be mutually discrepant in the past, they led to the complementarity principle, which is the basic principle underlying not only of quantum theory but also other domains of science and technology, and even applicable in everyday life.

As discrepancies are frequent phenomena in conceptual modeling, this positivist philosophy was introduced because, if relevantly and systematically accounted for, they may help to gather more information about the problem at hand. Following on with the reference to light, the wave and particle discrepancy triggered much of the research that ultimately clarified the field.

Note also that apparently, discrepant approaches can often lead to the same goal/problem solution. Let the following famous anecdote serve as an example. A physics teacher set the following question: show how it is possible to determine the height of a tall building with the aid of a barometer [46]. All the pupil's answers were all correct, although many of them did not use the approach—previously conceptualized and assumed—that the teacher was looking for based on the physical concepts to be assessed. The avoidance/concealment of discrepancies, as in this particular case, will obscure other approaches that might be even more practically feasible than the preferred option. This will lead to the dismissal of other options.

The policy of examining the importance of rather than killing discrepancies at the first sign is based on observations made in domains other than physics, like the sociology of organizations. This field considers that, in a broad sense, a discrepancy could be characterized as a key factor for intergroup communication, as well as productive work [47]. In requirements engineering, too, there exists empirical evidence in favor of considering discrepancies as factors that improve or guide the process of requirements acquisition and management [48], [49].

In any case, as suggested by Andrade *et al.* [44], the systematic "nonrejection" of discrepancies should not be a pretext for not solving them. In this respect, although conflicts and inconsistencies are both discrepancies, conflict solution and inconsistency solution require totally different approaches. Whereas stakeholders will have to reach an agreement and one or both parties will have to make some concession to solve a conflict, the solution of an inconsistency will require the correction of either of the discrepant elements, as, during conceptualization, one or both will not match reality. In other words, an inconsistency will require a process of refutation that should provide evidence of whether the information is true or false. To do this, the information should be classed as being present (as is) or potential (as should be).

Along these lines, a conceptual model will include both types of information, present and potential (desired/future). The as-is information can be derived directly from the problem domain, whereas the should-be information expresses concepts, aspects or behaviors. It is meaningless to check potential information against reality, as, by definition, it is a thing of the future. Remember that a conceptual model not only reflects the understanding of the problem, but also a possible solution to the problem. Changes of solution type, if desired, would imply the introduction of desired elements, something that is not provable or verifiable against the perceived reality.

A conceptual model should then effectively manage any type of discrepancies, which should be solved later by refutation or negotiation.

C. Methodology for Validating Holonic Structures

The goal is to specify a methodology for validating a holonic model m (holonic architecture) with its requirements $r_1, \ldots, r_n$. These requirements are the basis for building the model, that is to say, the trade needs that lead to the construction of the holonic system.

As mentioned earlier, this methodology should not be confined to only specific types of holons or implementation paradigms, but should be applicable to any requirements specification formalism that represents reality and provides a groundwork for building the model. The methodology aims to be general enough to accommodate both requirements types: unstructured natural language requirements and requirements formalized according to methods, like, for instance, i^*[50], or guidelines, like IEEE Std 830-1993 [51].

The proposed methodology is developed according to four iterative stages, which are detailed in the following.

Stage 1. Search for Minimal Submodels for Each Requirement: The minimal submodels $m_i, \ldots, m_j \subseteq m$ are determined for each $r_i (i = 1, \ldots, n)$, that is to say, the sets of roles played by holons within the holarchy, where:

$m' \subseteq m$ is a minimal model for $r_a \Leftrightarrow \forall m'' \subseteq m', m' \setminus m''$ (m' suppressed m'') is not a model for r_a

In other words, if there is a $r_i (i = 1, \ldots, n)$ and there is no $m' \subseteq m$ minimal model that represents r_i, then m is not a valid model.

A requirement may be represented by more than one minimal model. Consider, for instance, the "restocking management" requirement. A submodel of the model m accounts for this management by means of propagated restocking and another submodel denotes nonpropagated restocking, and both meet the requirement. This does not invalidate the model, but it may represent a discrepancy (conflict or inconsistency) that should be managed. According to the above example, $r_1 =$ "restocking management", m_1 is the m submodel for "restocking management by means of the propagated method" (where this functionality is known as r_t) and m_2 is the m submodel for "restocking management by means of the nonpropagated method" (where this functionality is known as r_s). In order to determine if there is a conflict, an inconsistency or a nondiscrepant situation where, however, modeling decisions are to be taken, we have to take a look at the real world, that is, the context of the requirements and of the model.

The options are the following.

1) The context determines that the propagated method (r_t) should be used for restocking management. Therefore, r_t is accepted, and it should be decided if r_s is.
2) The context determines that the nonpropagated method (r_s) should be used for restocking management. Therefore, r_s is accepted, and it should be decided if r_t is.
3) The context determines that both, r_t and r_s, are real information. This is an inconsistency that should be solved by refutation, that is to say, by searching how restocking is managed in the real world.
4) The context does not determine how restocking should be managed. Therefore, both r_t and r_s are potential information or desires. This is a conflict that should be

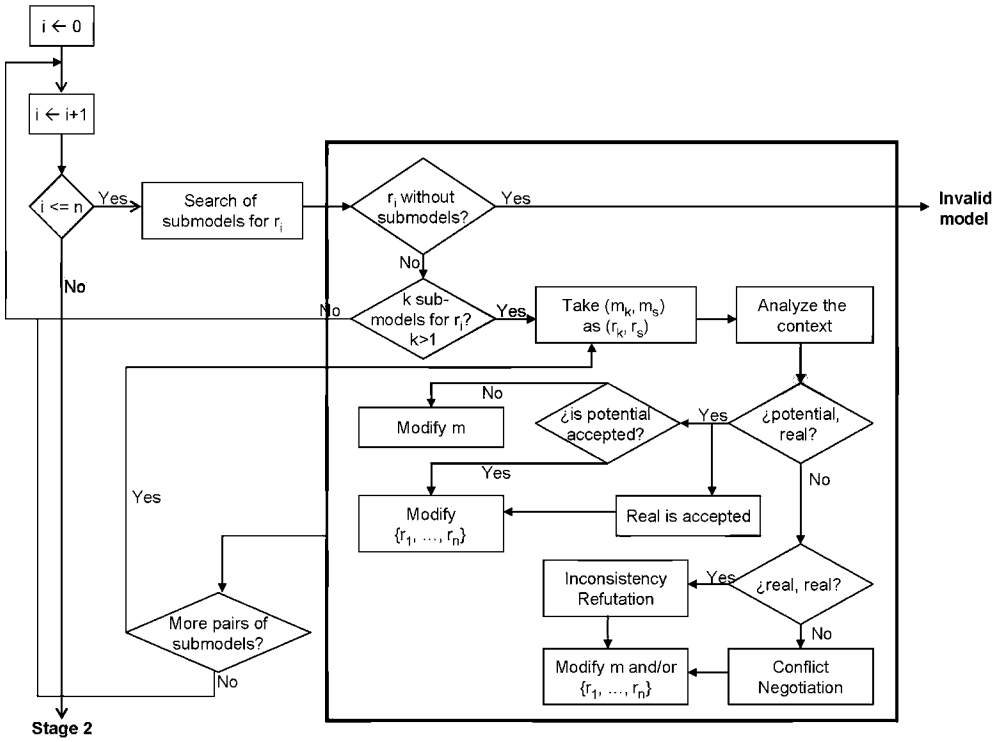

Fig. 1. Stage 1 of the validation process.

managed by negotiation, where either one, both or neither of the options for managing restocking will be accepted.

Each of the above situations has different implications regarding the requirements and/or the model. The implications are as follows.

1) Acceptance of r_t (or r_s) and rejection of r_s (or r_t), which implies:
   a) specifying r_1 in more detail, indicating the method for managing restocking agreed by consensus;
   b) building a new model m, where it is impossible to manage restocking using the method rejected by consensus;
   c) thoroughly validating the new model m.
2) Acceptance of r_t as well as of r_s, which implies:
   a) specifying r_1 in more detail, as the combination of r_t and r_s;
   b) continuing with stage 2 of the methodology without modifying the model m.
3) Rejection of r_t as well as of r_s, because it is finally decided either to manage restocking using a third method not included in the model or to remove the requirement. In this case, it is necessary to:
   a) Specify r_1—if it already exists—in more detail, indicating the method for restocking management agreed by consensus. If r_1 no longer exists, the original set of requirements representing reality should be modified.
   b) Build a new model m, where:
      i) There would be no option for restocking management using methods rejected by consensus.
      ii) In this case, there would exist a $m' \subseteq m$ for representing restocking management using this third method possibly agreed by consensus.
   c) Thoroughly validate the new model m again.

At the end of this stage—shown in Fig. 1—the model can be said to represent all the requirements; in other words, the model reflects reality. It remains to check whether the model only represents reality, that is, the set of real requirements and nothing other than these requirements. The remaining stages of the proposed methodology deal precisely with this point.

Stage 2. Search for Submodels That do not Represent any Requirement: If $m \setminus \{m_1, \ldots, m_j\} \neq \varnothing$ (m suppressed $\{m_1, \ldots, m_j\}$), that is to say, if $\cup_{i=1\ldots j} m_i \neq m$, where $\{m_1, \ldots, m_j\}$ is the group of minimal submodels of $r_1, \ldots, r_t$, $1 \leq n \leq t$ or $1 \leq t \leq n$ (as the original requirements could have been modified in stage 1) $\Rightarrow$ m has elements that represent a behavior not associated with any requirement.

These elements (parts of the model) may be related to other submodels (which supply inputs to or receive outputs from the elements) or be isolated elements. In any case, it would be necessary to:

1) study which new requirements are involved;
2) study whether there are any discrepancies (inconsistencies or conflicts) and, if so, manage them;
3) take the actions on the requirements and/or on the model, leading to either discrepancy solving or other decision making.

A couple of examples illustrate not only the potential situations—shown in Fig. 2—but also the dynamics of stage 2 as a whole.

Consider, when analyzing model m, that submodel m_1 represents requirement $r_1 =$ "restocking management by means of the propagated method" and also that submodel m_s enables "restocking management by means of the nonpropagated method" (known as r_s). As in stage 1, the context of both the

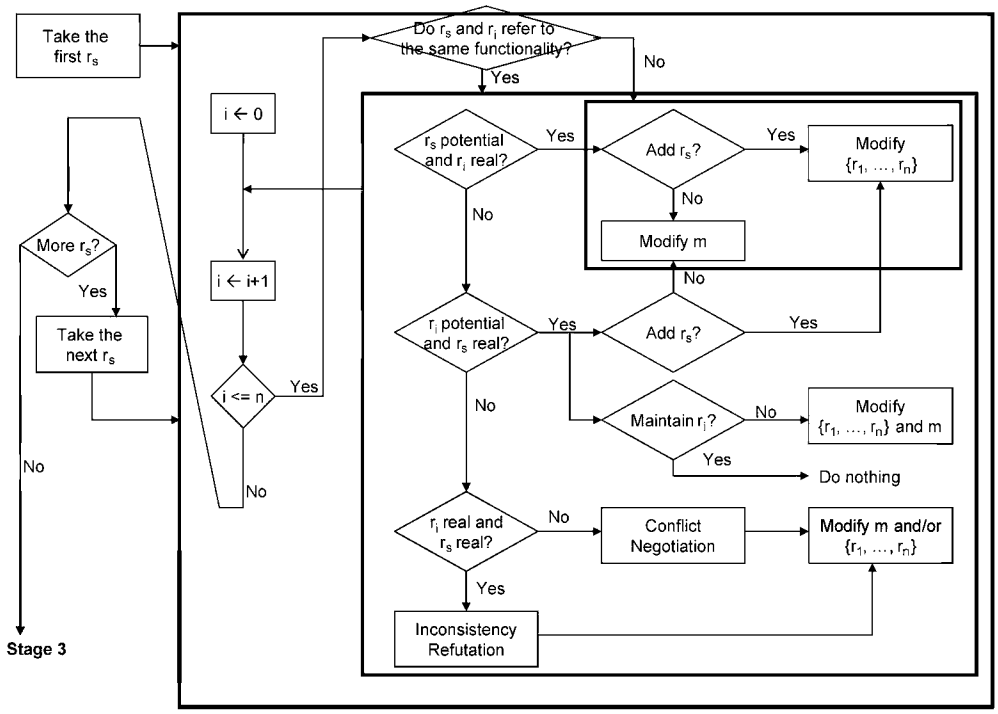

Fig. 2. Stage 2 of the validation process.

requirements and the model should be examined in order to determine if there is any discrepancy (conflict or inconsistency). The possible situations are:

1) The information represented by r_1 is real and the information represented by r_s is potential. There is no discrepancy. In this case, r_1 is simply accepted, and the stakeholders should decide whether r_s is added to the requirements.
2) The information represented by r_1 is potential, and the information represented by r_s is real. In this case, there is no discrepancy either. The stakeholders should simply decide whether to add r_s to the requirements, as well as whether to maintain r_1.
3) The information represented by both r_1 and r_s is real in the model context. In this case, there is an inconsistency: one of the requirements (r_1 or r_s) is wrong. The inconsistency should be solved by refutation; in other words, by finding out how restocking is really managed.
4) The information represented by both r_1 and r_s is potential. In this case, there is a conflict that should be solved by negotiation among the stakeholders in order to determine whether either, both or neither r_1 or r_s should be accepted.

Consider again that the requirement $r_1 =$ "restocking management by means of the propagated method" represents reality. We find, when analyzing model m, that submodel m_1 represents the requirement r_1, and there is also a new functionality not previously considered in the original requirements: a bad debtor register (r_s) performed by submodel m_s. There is no discrepancy between the original requirements (considering only r_1) and r_s, as they do not focus on the same functionality. There should be a simple discussion with the stakeholders to find out whether the functionality represented by r_s is desirable. Subsequently, r_s will be added to the original requirements or rejected by reducing the model, which will not then reflect this functionality.

Briefly, the analysis of model m during stage 2 may discover new functionalities not previously described in $r_1, \ldots, r_t$ requirements. These functionalities can lead to discrepancies in the shape of conflicts or inconsistencies or might not be discrepant. Discrepancy solving, as well as the decisions concerning nondiscrepant functionalities, implies deciding whether to include/exclude all or some of these functionalities in/from the requirements. This leads to the following situations:

1) If it is decided to include all the discovered functionalities, it would be necessary to:
   a) add the new functionalities to the set of original requirements;
   b) continue with stage 3 using the same model m.
2) If only some of the discovered functionalities are to be included, it would be necessary to:
   a) add the new functionalities to the set of original requirements;
   b) build a new model m based on the new set of requirements;
   c) validate the new model m according to the process explained in stage 1.
3) If none of the new functionalities is to be included, it would be necessary to:
   a) build a new reduced model m, omitting the elements representing the new functionalities;
   b) run the new model m through the validation process as of stage 1.

Stage 3. Search for Emerging Functionalities: If $\exists f$ functionality with $f \notin \{r_1, \ldots, r_x\}$, $1 \leq n \leq x$ or $1 \leq x \leq n$ (as the original requirements could have been modified during

stages 1 and 2) and $\exists m' \subseteq m$ with m' representing f, then m represents more than $r_1, \ldots, r_x$ due to emergent behavior.

In this case, as with the discovery of new functionalities described in stage 2, it would be necessary to detect whether the new situation is discrepant with the original requirements. If it is, the first thing to do is determine the type of discrepancy (conflict or inconsistency). In any case, the solution would imply either reducing the model—by removing f—or adding to the requirements—by including the functionality f as a new requirement in the original set. Fig. 2 gives an overview of this process, which has identical consequences to stage 2.

Stage 4 of the proposed methodology can be initiated with the new (or, if unchanged, the same) model m. This step should not be taken until all decisions and subsequent actions have been taken for each new functionality discovered and the possible effects on the requirements and/or the model have been addressed.

Stage 4. Search for Useless Behaviour in Holons/Holarchies: Let $\forall h \subseteq m$ and h be a holon that plays at least one role in the holonic model and $\forall$ (goal, abilities) $\in h$, then (goal, abilities) SR r_i with $i \in \{1, \ldots, y\}$, $1 \leq n \leq y$ or $1 \leq y \leq n$ must hold (as the original requirements might have changed in stages 1 to 3), where SR means "semantically related."

A goal is SR to a requirement if it is relevant for the satisfaction of the requirement in the model. In addition, if a goal is SR to a requirement, the requirement will be related to all its abilities by transitivity. For instance, consider the requirement $r_1 =$ "restocking management" and the holon h with the goal "calculate minimum stock". In this case, the goal of holon h (and, by transitivity, all its associated abilities) is usually SR to the requirement r_1. Other possible goals of holon h, like, for instance, "managing queues," would not usually be SR to the requirement r_1.

The previous expression can be also formulated as follows:

If $\exists h \subseteq m, h$ being a holon that plays at least one role in the holonic model and $\exists$ (goal, abilities) $\in h$ with (goal, abilities) $\neg$SR r_i with $i = 1 \ldots y$, $1 \leq n \leq y$ or $1 \leq y \leq n$ (as the original requirements might have changed in stages 1 to 3), then (goal, abilities) $\in h$ represents a behavior not initially proposed in requirements $r_1, \ldots, r_y$.

Again, as in stages 1 to 3, it would be necessary to determine whether the new situation creates a discrepancy with the original requirements. If it does, the first thing to do is determine the type of discrepancy (conflict or inconsistency). In any case, the solution would be either to reduce the model—by removing the goal and its abilities—or extend the requirements—by adding the functionality represented by the goal to the original set as a new requirement. Fig. 2 gives an overview of this process, as these options have identical consequences to stage 3.

A holonic model that successfully completes stage 4 of the proposed methodology can be said to be valid and acceptable as a reference model for the system to be developed and deployed.

Consequences of Model Changes: Special attention should be paid to the effects that a change to a model m has on the pre-existent model elements. This change could be a result of decisions made during any of the stages of the methodology proposed for the validation of holonic models:

1) What influence does the change have on simple holons?
   a) Do any holons disappear? This may happen either because the behavior that it represents is useless for the model requirements or because it has been subsumed by a new holon.
   b) Do any holons change? The goals and abilities of a holon are not static but change with the environment and, therefore, they may also change if the model is extended or reduced.
2) What influence does the change have on holarchies?
   a) Do any holarchies disappear? As with simple holons, a holarchy within a holonic model m may disappear if the behavior that it represents is useless for the model or if it has been subsumed by a new holarchy/holon. Note that the removal of a given holarchy also implies the removal of the roles that only it includes. It is important, on the other hand, not to remove the holons playing such roles, as the holons are independent elements: they exist independently of any role that they play.
   b) Do any holarchies change? The grounds for holon changes also apply to a given holarchy. A holarchy also has a set of roles that may change (creation of new roles, changes to or removal of existing roles). This implies re-applying the method for each role in order to find what holon/holarchy will play such role. Another situation that requires role reallocation is a change not of the role but of the holon playing the role. The change (modification or removal) of holons included in a holarchy of the holonic model m could make them either unsuitable for the roles that they play or simply disappear. In fact, a change or creation of any holon in the holonic model m should also lead to a role reallocation for every holarchy in the model, as the new/modified holon could then possibly be suitable for another (or the same) role in these holarchies.

The above changes, if any, could also imply other effects on the model (other possible changes) that should be dealt with. This process continues until there are no longer any effects on the model. This new model will undergo a new validation cycle starting from stage 1 of the proposed methodology.

Analysis of the Model: We have addressed the four stages of the proposed methodology for validating holonic models and discussed the practical guidelines for determining the effects of applying a change of the model on the remaining model. All these tasks imply the analysis of the model in order to discover new submodels, new functionalities and/or effects of changes.

Some of the most important techniques for model analysis—usually also known as validation techniques—proposed by different authors are the following.

1) Walk-through/one-step analysis [52]. The modeller might focus on different aspects of the model and even discover problems by specifying the model for another individual or group of individuals. Even if the target users do not

understand the details of the model, the modeller can perceive errors by simply studying the model in detail and trying to explain how it works. Preparation of model documentation could be similarly effective, as the modeller observes the model from a different viewpoint. In the absence of target users, the modeller should try to perform the same type of step-by-step analysis in order to gain an understanding of its behavior.

2) Model simplification [53]. Sometimes, a better understanding of the model can be gained by reducing the model to its minimal behavior, for example, a multiprocessor with only one processor to assure correct interaction between the processor and the shared memory or a PC-LAN with only two PCs. Once the simplified model has been built, it will useful for applying other techniques.
3) Other techniques, like tracing and model animation [54], can be used for simulation models, even using software packages like SimJava2 [55].

All these techniques, and any other considered applicable by the modeller or even designed ad hoc for a specific model, can be used to analyse holonic models during the validation process. A more efficient analysis would lead to a faster validation process, and it would be easier to get a critical response to the model validation questions.

III. Evaluation of the Proposal

Evaluation is objective proof that a method/process serves the intended purpose [56].

The efficiency of methodologies is, as a general rule, assessed by analyzing the results achieved after their execution in the real world. There is no way of specifying a number of tests cases that would guarantee the correctness of a given methodology, because any methodology might not be able to achieve the expected results for a problem new to the domain.

For this reason, we chose to stipulate the general-purpose criteria and properties that any methodology should meet. These criteria were mostly extracted from the mathematical field of logic, as, ultimately, any process or set of steps leading to a response can be expressed in terms of logic [57], [58].

Therefore, the parameters for consideration when evaluating a methodology are the following.

1) Credibility (fitness-for-purpose). Credibility refers to how the result of a process helps users to make correct decisions based on their original intention. It is a criterion measuring the practical usefulness of a process [59].
2) Traceability or decidability. A process is traceable or decidable when, for any given statement about the domain, there is a real path for determining whether the statement belongs to the set of domain truths.
3) Consistency. A process is consistent when it is not contradictory; meaning that a statement and its opposite are not simultaneously true.
4) Correctness. A process is correct if all the results that it considers to be true and false are indeed true and false, respectively.
5) Completeness. A set of steps fulfills the completeness criterion when it is sufficient to generate all the domain truths.

During the evaluation of these properties, remember not only that the domain of the proposed methodology consists of the holonic models built for any company, but that the goal of the methodology is to state the validity/invalidity of each of these models.

Consequently, this is direct proof of the credibility (fitness-for-purpose) of the methodology. In other words, the methodology empowers users to make correct decisions for their original intention—determine whether the holonic model is valid—, as it offers a critical response—positive or negative, with no probability rates—in all cases.

In order to prove traceability or decidability, we have to determine, on the one hand, the domain truths and, on the other, whether there is at least one path leading to those truths. We can state that there are two domain truths: 1) the model is invalid; and 2) the model is valid. The obvious path in the first case is that the model contains no submodel representing a requirement. In Fig. 1, which illustrates the first stage of the proposed methodology, this is the path that leads to the "Invalid model" output. There is not one but many paths to the valid model. To be precise, these are the paths that do not lead to the invalid model and where stakeholders have entered into acceptance agreements.

The consistency of the methodology is proven by the fact that there is no way of getting a positive and a negative response at the same time; that is to say, a model will always be valid or invalid, according to the proposed set of stages, but will never be valid and invalid at the same time.

The correctness of the methodology should be proved by guaranteeing that it does not return erroneous responses. The only case where the methodology states that a model is invalid is when it does not reflect all the original requirements. In no other case does the methodology reject the model, as, in a purist manner of speaking, the model is a—better or worse—representation of reality. In such cases, the methodology offers a set of guidelines for better aligning the model with reality or aligning the requirements with the model.

Last, it remains to test the completeness of the proposed methodology. This criterion is proven to be met by determining that the methodology is capable of generating all the truths in the domain. As mentioned previously, there are two domain truths: 1) the model is invalid; and 2) the model is valid. It obviously follows from the fulfillment of other criteria that the methodology is capable of returning either response, which proves its completeness.

IV. Case Study

Section III evaluates the methodology from the theoretical viewpoint. To assess the methodology from a practical viewpoint, this section presents an example of the methodology applied to validate a holonic model designed for a savings bank money laundering prevention system. This example was selected because it is a case study of a real application, to which

the general-purpose methodology is applicable, as it enables the validation of holonic models in any domain.

The appreciation of the importance of money laundering prevention has changed substantially from the conception there was some years ago in most countries. This is a result of a number of developments that have revealed the need to strengthen the money laundering prevention system all over the world and cooperate in the fight against criminal activities related to terrorism, drug trafficking, organized crime and other serious offenses.

The change that has taken place has led to a far-reaching reform of the regulations on the prevention of money laundering in most countries. In Spain, Act 19/1993 [60] on a selection of money laundering prevention measures was amended to transpose the new EU directive. Additionally, Act 12/2003 [61], in regard of the prevention of terrorist financing, was passed, and the additional rules are being revised or implemented.

The regulations on this matter are eminently preventive, which is a noteworthy point. The aim is to prevent funds from criminal activities from being channelled through the Spanish financial system and other sectors under obligation to prevent money laundering. Putting more and more obstacles in the way to block this channel will discourage actual criminal activities and, furthermore, safeguard the subjects governed by this act from the risks to which they are exposed by being used for laundering activities. In this respect, financial institutions will have to answer for any such activities that have been carried out through their organization. Therefore, they need to adopt two types of measures.

1) Measures aimed at detecting suspicious transactions before they are entered into to prevent the funds making their way into the system.
2) Measures that further scrutinize any suspicious transactions that have not been detected earlier, as this is the only way of gathering the knowledge needed to prevent this type of transactions from being entered into in the future.

"La Caixa" is now Spain's leading savings bank. With assets totalling over 439 200 million euros in 2011 and an extensive network of branches, composed of over 5200 offices, about 8000 cash dispensers (ATMs), over 28 000 employees and over 10 million customers, "La Caixa" has positioned itself as a top-notch institution and is now a benchmark for the Spanish financial sector. To support the monitoring of transactions likely to lead to money laundering, this institution has set up an information system, called the Integrated Money Laundering Prevention System (IMLPS). This system is integrated with La Caixa's conventional transactions systems, its branch network and connected to the Bank of Spain (BS), which is Spain's equivalent of the US Federal Reserve. Additionally, there are two levels of surveillance: the transaction, extending existing systematic reporting, and the customer, centrally analyzing what information branches have about customers and making use of customer information.

Furthermore, the system includes a process that manages internal and external knowhow, scattered across and outside the organization. On the one hand, this process prevents redundancies and duplicated effort and, on the other, it assures that knowledge will not be lost if the people who have it leave the company.

The system operational flow is as follows.

1) La Caixa's IMLPS is based at the Laundering Prevention Operations Unit (LPOU), which is responsible for monitoring and reporting suspicious transactions and customers to the BS.
2) The system generates alerts that should be checked by the LPOU and by the branches. The LPOU checks alerts, and the branches make a decision on any alerts that they receive. The branches inspect the alerts received by means of an application integrated in the financial terminal.
3) When an irregularity is detected, the branches generate a document that formalizes the report in a manner that is totally integrated into the application. This document is also used in the event of spontaneous reports by branches in the event of suspicious transactions.
4) The LPOU is the unit responsible for setting the system operating parameters.

The IMPLS conceptual architecture is based on a holonic model. For this reason, the system has been taken as an example of the application of the holonic conceptual model validation methodology presented here. On size grounds, however, we examine a subset of this architecture that is considered representative enough to illustrate the four phases of the proposed methodology. In this case, we look at customer monitoring, and its immediate consequences.

Customer monitoring has proven to be one of the key points in money laundering detection. Its operation is based on customer "surveillance" and on the generation of alerts that should be checked by the LPOU or the respective branches. When an alert has sounded, the customer's branch will have to examine the alert to either except it or press ahead with the inspection and report to the LPOU. Exception means excluding customers about which enough information is available to presume that they do not enter into transactions that are classifiable as laundering from the customer analysis system. Excepted customers are regular customers with known lawful activities that are excepted for a renewable period. Exceptable customers are typically hypermarkets, petrol stations, local government, etc.

For this example, the original requirements for the construction of the holonic model of the selected subset of IMPLS functionalities are:

r_1: "Detect customers that are characteristically inclined to or at risk of money laundering."
r_2: "Report suspect customers to offices."
r_3: "Except suspect customers or pass on alert to LPOU for solution."

Fig. 3 shows the holarchy designed to satisfy these requirements.

The Hsystem holarchy represents the holonic model m and is composed of three roles R1, R2, and R3. Each of these roles is played by the holarchies H1, H2, and H3, respectively. Each holarchy again defines its own set of roles. Thus, H1 defines

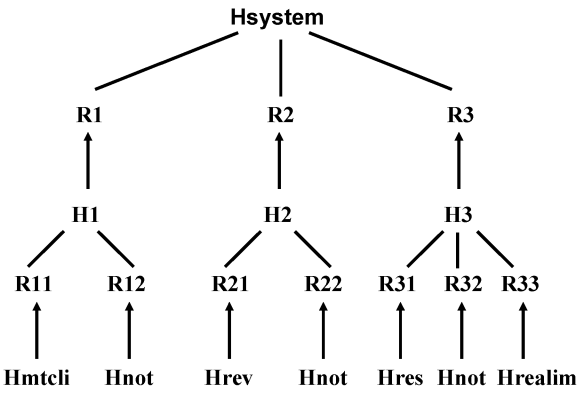

Fig. 3. IMLPS Holarchy.

TABLE II
ROLE DESCRIPTION

Role Name		Description
R1		Monitoring and analysis of customer information from operational systems.
	R11	Monitor customer.
	R12	Report suspect customer alert to respective branch.
R2		Inspect alert at branch.
	R21	Except customer or enter inspection results and any other observations in the information base.
	R22	Report the result of the branch alert inspection to LPOU.
R3		Inspect alert at LPOU.
	R31	Process alert report to solve incident and define means of action.
	R32	Report alert solution to respective branch.
	R33	Store applicable historical data.

roles R11 and R12 played, respectively, by holons Hmtcli and Hnot. H2 is composed of roles R21 and R22, played by holons Hrev and Hnot. Finally, holarchy H3 is composed of roles R31, R32, and R33 played by holons Hres, Hnot, and Hrealim.

Tables II and III describe each role and each holon, respectively. The key descriptors are Name and Description for roles and Name, Description, Goals, and Abilities for holons. The other descriptors (Identifier, Synonyms, Abbreviations, Support sources, and Observations) are useful for identifying and further detailing the element, but the methodology is applicable even if they are not specifically stated.

Let us now apply the stages of the methodology to validate the holonic model m (holonic architecture) with requirements r_1, r_2, and r_3, that is, guarantee that the holonic model matches no other than the real requirements.

Stage 1. Search for Submodels for Each Requirement

The identified minimum submodels for each requirement are:
$m_1 = \{(\text{R11}, \text{Hmtcli})\}$ minimal submodel for r_1.
$m_2 = \{(\text{R12}, \text{Hnot})\}$ minimal submodel for r_2.
$m_3 = \{(\text{R2}, \text{H2})\}$ minimal submodel for r_3.

All the requirements are represented in a minimal submodel, and there is only one minimal submodel for each requirement. Therefore, the model passes the first stage of the validation, that is, all the requirements can be said to be represented in the model. In other words, the model represents reality. Let us now check that model represents nothing other than reality, meaning

TABLE III
HOLON/HOLARCHY DESCRIPTION

Holon Name	Descriptors	
Hmtcli	Description	Determine whether customer information from the company's transaction systems is suspect of money laundering.
	Goals	Analyse information through the frames system.
	Abilities	Analysis (A). Decision making (DM). Learning (L).
Hnot	Description	Holon reporting input information to target entity.
	Goals	Report information to BS. Report information to branch network. Report information to LPOU.
	Abilities	Secure communications (SC). Correct reception control (CRC). Communication failure recovery (CFR).
Hrev	Description	Holon analysing or inspecting an alert to except the customer and/or store the results of the inspection, as well as any observations in the information base.
	Goals	Except suspect customers. Store new data in the information base.
	Abilities	Analysis (A). Decision making (DM). Failure recovery (FR). Update control (UC).
Hres	Description	Holon processing an alert report to define the means of action.
	Goals	Solve alert report using frames system. Define means of action using scripts system.
	Abilities	Analysis (A). Decision making (DM). Learning (L).
Hrealim	Description	Holon entering information in the information base.
	Goals	Validate alert reports using induction techniques. Enter new data in the information base.
	Abilities	Analysis (A). Decision making (DM). Failure recovery (FR). Update control (UC).

its set of requirements and no other than these requirements. This is the purpose of the other stages of the methodology.

Stage 2. Search for Submodels That do not Represent any Requirement

Note that $m \setminus \{m_1, m_2, m_3\} = \{(\text{R3}, \text{H3})\} \neq \varnothing$, that is to say, $\cup_{i=1\ldots3} m_i \neq m$.

The submodel $m' = \{(\text{R3}, \text{H3})\}$ represents a behavior that is not associated with any requirement.

According to the methodology, the first step is to study which other requirements are involved. This way, we discover the following requirements:

r_4: "Solve alerts and report to respective branch."
r_5: "Store historical information on generated and solved alerts in the system."

Now we have to examine whether there are any discrepancies (inconsistencies or conflicts) and, if so, deal with them. There is no discrepancy between the original requirements and the new

requirements, as they do not refer to the same functionalities. Therefore, we merely have to discuss with the stakeholders whether the functionalities represented by r_4 and r_5 are desirable. In this case, both functionalities are desirable, meaning that r_4 and r_5 will be added to the original requirements. This way, the new set of requirements will be as follows:

r_1: "Detect customers that are characteristically inclined or at risk of money laundering."
r_2: "Report suspect customers to respective branch."
r_3: "Except suspect customers or report alert to LPOU for solution."
r_4: "Solve alerts and report to respective branch."
r_5: "Store historical information on generated and solved alerts in the system."

The model is unchanged, and we can then move on stage 3 using the same model m.

Stage 3. Search for Emerging Functionalities

As defined in the model, there is an option for reporting the alert inspection result for an excepted customer to the LPOU. However, alert management at the LPOU only makes sense for alerts concerning nonexcepted customers as it involves the consideration, solution, and definition of means of action.

Accordingly, let $\exists f =$ "Report alerts concerning excepted customers to LPOU" functionality with $f \notin \{r_1, r_2, r_3, r_4, r_5\}$ and $m' = \{(\text{R22}, \text{Hnot})\} \subseteq m$ with m' representing f, then m represents more than r_1, r_2, r_3, r_4, r_5 due to emergent behavior.

Then, we have to detect whether the new situation leads to a discrepancy with the original requirements. In this case, the new functionality may be discrepant with r_3, as they both refer to the same functionality. To determine whether there really is a discrepancy (conflict or inconsistency), it will be necessary to examine the real world to determine whether the information present in both r_3 and in f (say r_6) is real or potential.

Requirement r_3 refers to the exception of suspect customers or to the reporting of alerts to LPOU for solution. Therefore, it is possible to infer from r_3 that only alerts from nonexcepted customers are sent to the LPOU. In this case, r_3 represents the real information, as only alerts belonging to nonexcepted customers are sent to the LPOU in the real banking institution operations. In the knowledge that r_3 is real and r_6 is potential, we conclude that there is no discrepancy, and it remains to determine whether r_6 represents a desirable functionality. After discussion with the stakeholders, it is decided that the system should not offer the option of reporting alerts concerning excepted customers to the LPOU, meaning that r_6 is not desirable. This way, requirement r_3 is maintained, and the functionality represented by r_6 will have to be removed from the model. Consequently, role R22 will have to be redefined as "Report the result of the branch alert inspection to the LPOU if the customer has not been excepted." The remainder of the model is unchanged.

If any change is made to the model, the validation process has to be reapplied to assure the changes have not generated any side-effects. Therefore, the new model has to be revalidated from stage 1:

— Stage 1. Search for submodels for each requirement.
$m_1 = \{(\text{R11}, \text{Hmtcli})\}$ minimal submodel for r_1.
$m_2 = \{(\text{R12}, \text{Hnot})\}$ minimal submodel for r_2.
$m_3 = \{(\text{R2}, \text{H2})\}$ minimal submodel for r_3.
$m_4 = \{(\text{R31}, \text{Hres}), (\text{R32}, \text{Hnot})\}$ minimal submodel for r_4.
$m_5 = \{(\text{R33}, \text{Hrealim})$ minimal submodel for r_5.
All the requirements are represented by a minimal submodel, and there is only one minimal submodel for each requirement. Therefore, the model passes the first validation phase, and we can move on to stage 2.
— Stage 2. Search for submodels that do not represent any requirement. With the new situation, note that $m \setminus \{m_1, m_2, m_3, m_4, m_5\} = \varnothing$, that is to say, $\cup_{i=1\ldots5}\, m_i = m$. Therefore, there are no submodels in the general model that do not represent any requirement, and the model passes this stage.
— Stage 3. Search for emerging functionalities. In this case, no new emerging functionalities are detected, and we move on to the next stage of the methodology without making any change to the model and requirements.

Stage 4. Search for Useless Behaviour in Holons/Holarchies

In this case, one of the goals of the holon Hnot is a behavior that is never used in the model. This behavior is report to the BS.

Therefore, $\exists h = \text{Hnot} \subseteq m$, which plays roles R12, R22, and R32 and $\exists(\text{goal, abilities}) =$ ("Report to BS", (SC, CRC, CFR)) $\in$ Hnot is useless behavior.

Again, we have to determine whether the new situation creates discrepancy with the original requirements and take the proper steps.

Reporting is covered in the original requirements by r_2 and r_4. The new behavior (say r_6) is, therefore, potentially discrepant with those requirements. To determine whether such a discrepancy exists, we have to investigate the context of both the requirements and the model; that is, examine the real world.

Going back to the reality of money laundering, we find that both the reporting to which requirements r_2 and r_4 refer and the reporting represented by r_6 are real, as they take place in the context. Therefore, there is a discrepancy and, more specifically, an inconsistency that will have to be solved by refutation. To do this, we have to focus on the subset of reality that represents the reality with which we are concerned, that is, the part of reality that this holonic model represents. In this case, the focus is on customer monitoring, together with its immediate consequences. In this context, r_6 is refuted, and it will have to be removed from the model. The behavior represented by r_6 will have to be added to the requirements and to the model to model the reality of communication with the BS, but at the moment it is useless. Consequently, the goal "Report to the BS" is removed from the holon Hnot. On the other hand, r_2 and r_4 are kept as requirements.

The model has been changed, meaning that it has to go through the entire validation process again, as specified below:

— *Stage 1. Search for submodels for each requirement.*
$m_1 = \{(\text{R11}, \text{Hmtcli})\}$ minimal submodel for r_1.

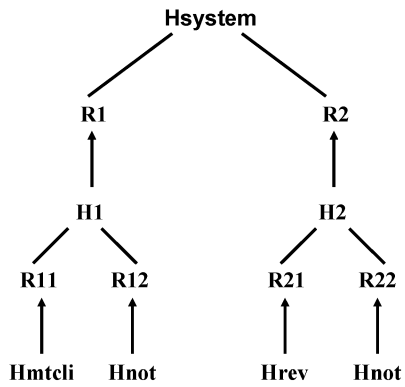

Fig. 4. IMLPS Holarchy (new version).

$m_2 = \{(\text{R12}, \text{Hnot})\}$ minimal submodel for r_2.
$m_3 = \{(\text{R2}, \text{H2})\}$ minimal submodel for r_3.
$m_4 = \{(\text{R31}, \text{Hres}), (\text{R32}, \text{Hnot})\}$ minimal submodel for r_4.
$m_5 = \{(\text{R33}, \text{Hrealim})\}$ minimal submodel for r_5.

All the requirements are represented by a minimal submodel, and there is only one minimal submodel for each requirement. Therefore, the model passes the first validation phase, and we can move on to stage 2.

— *Stage 2. Search for submodels that do not represent any requirement.* In this case $m \backslash \{m_1, m_2, m_3, m_4, m_5\} = \varnothing$, that is, $\cup_{i=1...5} m_i = m$. Therefore, the general model contains no submodels that do not represent any requirement and the model passes this stage.

— *Stage 3. Search for emerging functionalities*. No new emerging functionalities are detected, and we move on to the next stage of the methodology without making any change to the model and requirements.

— *Stage 4. Search for useless behavior in holons/ holarchies*. With the new situation, no other useless behavior is discovered in holons/holarchies, meaning the model passes stage 4 of the methodology.

At this point, the holonic model can be said to represent the set of requirements and nothing other than those requirements. Thus, the model is valid, and it is accepted as a reference model for the system to be developed and deployed.

In the following, we give an example of an invalid model, built on the requirements:

r_1: "Detect customers that are characteristically inclined to or at risk of money laundering."
r_2: "Report suspect customers to branches."
r_3: "Except suspect customers or report alert to LPOU for solution."
r_4: "Solve alerts and report to respective branch."
r_5: "Store historical information on the generated and solved alerts in the system."

Fig. 4 shows the holonic model designed to meet these requirements.

The definitions and characteristics of the roles and holons in the holarchy are as specified in Tables II and III.

Again, the proposed methodology is applied to determine whether the above holonic model is valid.

Stage 1. Search for Submodels for Each Requirement

The minimal submodels identified for each requirement are as follows:

$m_1 = \{(\text{R11}, \text{Hmtcli})\}$ minimal submodel for r_1.
$m_2 = \{(\text{R12}, \text{Hnot})\}$ minimal submodel for r_2.
$m_3 = \{(\text{R2}, \text{H2})\}$ minimal submodel for r_3.
$m_4 = \{\}$ minimal submodel for r_4.
$m_5 = \{\}$ minimal submodel for r_5.

Clearly, there are no minimal submodels for either r_4 or r_5. In other words:

There is r_4 and there is no $m' \subseteq m$ minimal model that represents r_4.
There is r_5 and there is no $m' \subseteq m$ minimal model that represents r_5.
Then m is not a valid model.

V. Conclusion

This paper proposes a methodology for validating holonic models based on the trade requirements for their construction. No matter how the models are implemented, a given holonic architecture is a model that then needs to be validated against reality in order to guarantee its quality before it is implemented and deployed.

As a general rule, the literature related to validation mechanisms demonstrates how algorithms work rather than examining the quality of the holonic architecture; namely, it does not primarily focus on verifying whether the models meet the requirements of reality and nothing other than these requirements. Such validation is, however, crucial as—in organization-wide terms—the efficiency of a holonic system depends on this.

Along these lines, the holonic model must satisfy the requirements for which it is built, but it is equally important that it should satisfy only that set of requirements. This is because the organization does not have unlimited time, finances, and resources, and they should be properly managed. Therefore, when building a holonic system to satisfy certain trade requirements, it is necessary to check that no relationships involving more responsibilities than are strictly necessary emerge, as they will usually also involve unnecessary time, cost, and/or resources.

The methodology proposed here addresses the validation of holonic models from this viewpoint. The proposal is not restricted to any specific type of holons or any specific implementation paradigm. Neither are there any restrictions on the domain and the formalisms for specifying the requirements that represent reality and that are the basis of the model. The methodology's main strength is that it is a general-purpose approach, as it will not be invalidated by new holon types, new implementation paradigms for holonic systems and new specification methods from emerging. This generality has been achieved, on the one hand, by borrowing concepts from the area of conceptual modeling and, on the other, by applying the pure concepts of holon and holarchy, shunning any specific implementation paradigm, as well as any specific set of holon types.

REFERENCES

[1] H. Van Brussel, L. Bongaerts, J. Wyns, P. Valckenaers, and T. Van Ginderachter, "A conceptual framework for holonic manufacturing: Identification of manufacturing holons," *J. Manuf. Syst.*, vol. 18, no. 1, pp. 35–52, 1999.

[2] M. M. T. Giebels, H. J. J. Kals, and W. H. M. Zijm, "Building holarchies for concurrent manufacturing planning and control in EtoPlan," *Comput. Ind.*, vol. 46, no. 3, pp. 301–314, Oct. 2001.

[3] A. T. Jones and C. R. McLean, "A proposed hierarchical control model for automated manufacturing systems," *J. Manuf. Syst.*, vol. 5, no. 1, pp. 15–26, 1986.

[4] D. M. Dilts, N. P. Boyd, and H. H. Whorms, "The evolution of control architectures for automated manufacturing systems," *J. Manuf. Syst.*, vol. 10, no. 1, pp. 79–93, 1991.

[5] B. Huang, H. Gou, W. Liu, Y. Li, and M. Xie, "A framework for virtual enterprise control with the holonic manufacturing paradigm," *Comput. Ind.*, vol. 49, no. 3, pp. 299–310, Dec. 2002.

[6] G. N. Franco and A. Botacchio, "The holonic paradigm as a new metaphor for the coordination problem of virtual enterprises," in *Proc. 2nd IFIP Working Conf. Infrastruct. Virtual Org.*, Florianópolis, Brazil, 2000.

[7] H. Paggi and F. Alonso, "A holonic model of system for the resolution of incidents in the software engineering projects," in *Proc. Int. Conf. Comput. Autom. Eng.*, Bangkok, Thailand, 2009, pp. 79–86.

[8] E. F. Tagne, F. Lauri, A. Koukam, and E. Tonye, "The coverage problem in wireless sensor networks by holonic multi-agent approach," *Int. J. Comput. ICT Res.*, vol. 3, no. 1, pp. 32–41, 2009.

[9] A. R. Hilal, A. Khamis, and O. Basir, "A holonic federated sensor management framework for pervasive surveillance systems," in *Proc. IEEE Int. Syst. Conf.*, Montreal, QC, Canada, 2011, pp. 361–366.

[10] Y. Ye, V. Hilaire, A. Koukam, and C. Wandong, "A holonic model in wireless sensor networks," in *Proc. Int. Conf. Intel. Inf. Hiding Multimedia Signal Process.*, Harbin, China, 2008, pp. 491–495.

[11] A. Koestler, *The Ghost in the Machine*. London, U.K.: Hutchinson, 1967.

[12] H. Lawson-Tancred, *Aristotle: Metaphysics*. London, U.K.: Penguin, 1985.

[13] H. A. Simon, "The architecture of complexity," *Proc. Amer. Philosoph. Soc.*, vol. 106, no. 6, pp. 467–482, Dec. 1962.

[14] A. Giret and V. Botti, "Engineering holonic manufacturing systems," *Comput. Ind.*, vol. 60, no. 6, pp. 428–440, Aug. 2009.

[15] P. Valckenaers, H. Van Brussel, J. Wyns, L. Bongaerts, and P. Peeters, "Designing holonic manufacturing systems," *Robot. Comput.-Integr. Manuf.*, vol. 14, no. 5/6, pp. 455–464, Oct. 1998.

[16] H. Van Brussel, J. Wyns, P. Valckenaers, and P. Peeters, "Reference architecture for holonic manufacturing systems: PROSA," *Comput. Ind.*, vol. 37, no. 3, pp. 255–274, Nov. 1998.

[17] P. Leitao, A. W. Colombo, and M. Restivo, "ADACOR: A collaborative production automation and control architecture," *IEEE Intel. Syst.*, vol. 20, no. 1, pp. 58–66, Jan./Feb. 2005.

[18] L. Gou, P. B. Luh, and Y. Kyoya, "Holonic manufacturing scheduling: Architecture, cooperation mechanism, implementation," *Comput. Ind.*, vol. 37, no. 3, pp. 213–231, Nov. 1998.

[19] W. Shen, F. Maturana, and D. H. Morrie, "MetaMorph II: An agent-based architecture for distributed intelligent design and manufacturing," *J. Intel. Manuf.*, vol. 11, no. 3, pp. 237–251, Jun. 2000.

[20] D. Kotak, S. Wu, M. Fleetwood, and H. Tamoto, "Agent-based holonic design and operations environment for distributed manufacturing," *Comput. Ind.*, vol. 52, no. 2, pp. 95–108, Oct. 2003.

[21] P. Blanc, I. Demongodin, and P. Castagna, "A holonic approach for manufacturing execution system design: An industrial application," *Eng. Appl. Artif. Intel.*, vol. 21, no. 3, pp. 315–330, Apr. 2008.

[22] B. Clegg, "Building a holarchy using business process orientated holonic PrOH modeling," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 37, no. 1, pp. 23–40, Jan. 2007.

[23] Y. Feili, T. Fang, and K. R. Pattipati, "Integration of a holonic organizational control architecture and multiobjective evolutionary algorithm for flexible distributed scheduling," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 38, no. 5, pp. 1001–1017, Sep. 2008.

[24] J. M. Simao, C. A. Tacla, and P. C. Stadzisz, "Holonic control metamodel," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 39, no. 5, pp. 1126–1139, Sep. 2009.

[25] S. A. Peterson, M. Divitini, and M. Matskin, "An agent-based approach to modelling virtual enterprises," *Prod. Planning Control*, vol. 12, no. 3, pp. 224–233, 2001.

[26] P. Leitao and F. J. Restivo, "Implementation of a holonic control system in a flexible manufacturing system," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 38, no. 5, pp. 699–709, Oct. 2008.

[27] S. S. Heragu, R. J. Graves, B. Kim, and A. St. Onge, "A. Intelligent agent based framework for manufacturing systems control," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 32, no. 5, pp. 560–573, Sep. 2002.

[28] A. Giret and V. Botti, "Holon and agents," *J. Intel. Manuf.*, vol. 15, no. 5, pp. 645–659, Oct. 2004.

[29] K. Fischer, M. Schillo, and J. Siekmann, "Holonic multi-agent systems: A foundation for organisation of multiagent systems," in *Proc. LNAI*, 2003, vol. 2744, pp. 71–80.

[30] M. Matskin, M. Divitini, and S. A. Petersen, "An architecture for multi-agent support in a distributed information technology application," in *Proc. KI*, A. Holsten, G. Joeris, C. Klauck, M. Klush, H.-J. Muller, and J. P. Muller, Eds., Bremen, Germany, 1998, pp. 47–58.

[31] M. Matskin, "Multi-agent support for modeling co-operative work," in *Intelligence in Networks*, T. Yongchareon, F. A. Aagesen, and V. Wuwongse, Eds. Dordrecht, The Netherlands: Kluwer, 1999, pp. 419–432.

[32] R. G. Smith, "The contract net protocol: High-level communications and control in a distributed problem solver," *IEEE Trans. Comput.*, vol. C-29, no. 12, pp. 1104–1113, Dec. 1980.

[33] J. Andrade, J. Ares, R. García, J. Pazos, S. Rodríguez, and A. Silva, "A methodological framework for generic conceptualisation: Problem-sensitivity in software engineering," *Inf. Softw. Technol.*, vol. 46, no. 10, pp. 635–649, Aug. 2004.

[34] P. Leitao, A. W. Colombo, and F. Restivo, "A formal validation approach for holonic control system specifications," in *Proc. 9th Int. Conf. Eng. Technol. Factory Autom.*, Lisbon, Portugal, 2003, pp. 203–210.

[35] M. Bal and M. Hashemipour, "Virtual factory approach for implementation of holonic control in industrial applications: A case study in die-casting industry," *Robot. Comput.-Integr. Manuf.*, vol. 25, no. 3, pp. 570–581, Jun. 2009.

[36] Java Agent DEvelopment Framework. [Online]. Available: http://jade.tilab.com/

[37] A. W. Colombo, R. Carelli, and B. Kuchen, "A temporised Petri net approach for design, modelling and analysis of flexible production systems," *Int. J. Adv. Manuf. Technol.*, vol. 13, no. 3, pp. 214–226, Mar. 1997.

[38] A. W. Colombo, "Development and implementation of hierarchical control structures of flexible production systems using high-level petri nets," in *Manufacturing Automation Series*, K. Feldmann, Ed. Bamberg, Germany: Meisenbach-Verlag, 1998.

[39] P. Checkland, *Soft Systems Methodology in Action*. Chichester, U.K.: Wiley, 1999.

[40] C. Lévy-Leboyer, "Selection and assessment in Europe," in *Handbook of Industrial and Organizational Psychology*, H. C. Triandis, M. D. Dunnette, and L. M. Hough, Eds. Palo Alto, CA: Consulting Psychol. Press, 1990.

[41] L. L. Thurstone and T. G. Thurstone, *Aptitudes Mentales Primarias [Primary Mental Abilities]*. Madrid, Spain: TEA, 2002.

[42] N. Anderson, D. S. Ones, H. K. Sinangil, and C. Viswesvaran, *Handbook of Industrial, Work and Organizational Psychology Personnel Psychology*. Thousand Oaks, CA: Sage Pub. Ltd., 2002.

[43] J. L. Soriano, "Architectural Model for the Cooperative Management of Distributed Systems and Services Based on Autonomous Agents," Ph.D. dissertation, Technical Univ. Madrid, Madrid, Spain, 2003.

[44] J. Andrade, J. Ares, R. García, J. Pazos, S. Rodríguez, and A. Silva, "A methodological framework for viewpoint-oriented conceptual modeling," *IEEE Trans. Softw. Eng.*, vol. 30, no. 5, pp. 282–294, May 2004.

[45] A. Calandra, "The History of the Barometer," in *The Project Physics Course*. New York: Holt, Rinehart and Winston, 1975.

[46] A. Silva, "Requirements, domain and specifications: A viewpoint-based approach to requirements engineering," in *Proc. ICSE*, 2002, pp. 94–104.

[47] S. Robbins, *Organizational Behavior: Concepts, Controversies, Applications*. Englewood Cliffs, NJ: Prentice-Hall, 2000.

[48] S. Bendifallah and W. Scacchi, "Work structures and shifts, an empirical analysis of software specification teamwork," in *Proc. 11th ICSE*. New York: ACM Press, 1989, pp. 260–270.

[49] W. Robinson, "Negotiation behavior during requirements specification," in *Proc. 12th ICSE*, 1990, pp. 268–276.
[50] G. Grau, X. Franch, and N. Maiden, "PRiM: An i^*-based process reengineering method for information systems specification," *Inf. Softw. Technol.*, vol. 50, no. 1/2, pp. 76–100, Jan. 2008.
[51] *IEEE Recommended Practise for Software Requirements Specification*, IEEE Std 830-1993.
[52] E. Yourdon, *Structured Walkthroughs*. Englewood Cliffs, NJ: Prentice-Hall, 1989.
[53] Software Engineering Institute, Model-Based Verification. A Technology for Dependable System Upgrade 1998. [Online]. Available: http://www.sei.cmu.edu/pub/documents/98.reports/pdf/98tr009.pdf
[54] J. P. C. Kleijnen, "Verification and validation of simulation models," *Eur. J. Oper. Res.*, vol. 82, no. 1, pp. 145–162, Apr. 1995.
[55] SimJava. [Online]. Available: http://www.icsa.inf.ed.ac.uk/research/groups/hase/simjava/
[56] International Organization for Standardization, ISO Guide 8402: Quality-Vocabulary, Genève, Switzerland 1994.
[57] A. Tarski, *Introducción a la lógica*. Madrid, Spain: Espasa-Calpe, 1951.
[58] B. Russell and A. N. Whitehead, *Principia mathematica*. Madrid, Spain: Paraninfo, 1981.
[59] R. Boqué, A. Maroto, J. Riu, and X. Rius, "Validation of analytical methods," *Grasas y Aceites*, vol. 53, no. 1, pp. 128–143, 2002.
[60] Spanish Official Journal (Boletín Oficial del Estado). [Online]. Available: http://www.boe.es/boe/dias/2003-05-22/pdfs/A19490-19494.pdf
[61] Spanish Official Journal (Boletín Oficial del Estado). [Online]. Available: http://www.boe.es/boe/dias/2005_01-22/pdfs/A02573-02583.pdf